Booting xtensa linux on qemu

Issues

wrong address generated for jump in kernel_exception_return

arch/xtensa/kernel/entry.S:777

```
        movi    a0, 1f + (0x40000000 - 0xC0000000)      # Calculate Return
address for "1f" and store in a0
        rsil    a2, XCHAL_EXCM_LEVEL    # FIXME: again, only do this if
PS.INTLEVEL <= EXCM_LEVEL
        retw                            # rotate back by 4 registers,
possibly with underflow
                                        # Back out our _entry Frame above...

                                        # ... We return here from above
retw.
1:      mov     a1, a5                  # our a1 became a5 at the above
_entry, a1 = a5
```

Assembled it looks like this:

```
0010 ffffff3f 00000000 000000c0 05030080
...
                        1c: R_XTENSA_32 .text
...
 2fd:   000001          l32r    a0, fffc0300 <ret_from_fork+0xfffbf858>
                        2fd: R_XTENSA_SLOT0_OP  .literal+0x1c
 300:   006320          rsil    a2, 3
 303:   f01d            retw.n
 305:   051d            mov.n   a1, a5
```

Linked vmlinux looks like this (broken, word at d000357c should be 2d3d0050):

```
d0003570 ffffff3f e83b00d0 000000c0 cd3c0050
...
d0003d25:       fe1501          l32r    a0, d000357c <T$339+0x68>
d0003d28:       006320          rsil    a2, 3
d0003d2b:       f01d            retw.n
d0003d2d:       051d            mov.n   a1, a5
```

Looks like bug in linker. Cured by the following patch:

```
diff --git a/arch/xtensa/kernel/entry.S b/arch/xtensa/kernel/entry.S
index 0cb1530..e7b2263 100644
--- a/arch/xtensa/kernel/entry.S
```

```
+++ b/arch/xtensa/kernel/entry.S
@@ -774,7 +774,9 @@ _kernel_exception:
 #endif

        l32i    a3, a1, PT_PS                           # a3 = ptregs->ps
[NOTE: Used below after retw]
-       movi    a0, 1f + (0x40000000 - 0xC0000000)      # Calculate Return
address for "1f" and store in a0
+       movi    a0, 1f # + (0x40000000 - 0xC0000000)    # Calculate Return
address for "1f" and store in a0
+       movi    a2, 0x40000000 - 0xC0000000
+       add     a0, a0, a2
        rsil    a2, XCHAL_EXCM_LEVEL    # FIXME: again, only do this if
PS.INTLEVEL <= EXCM_LEVEL
        retw                            # rotate back by 4 registers,
possibly with underflow
                                        # Back out our _entry Frame above...
```